UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/051,274	01/22/2002	Sunil Kunisetty	50277-2895	7586

42425 7590 03/25/2010

HICKMAN PALERMO TRUONG & BECKER/ORACLE
2055 GATEWAY PLACE
SUITE 550
SAN JOSE, CA 95110-1083

EXAMINER	
PHAM, CHRYSTINE	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
03/25/2010	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte SUNIL KUNISETTY, JULIE BASU,
and KWOK LUN ALEX YIU

Appeal 2009-002783
Application 10/051,274
Technology Center 2100

Decided: March 25, 2010

Before JOHN A. JEFFERY, JAMES D. THOMAS, and
LANCE LEONARD BARRY, *Administrative Patent Judges*.

BARRY, *Administrative Patent Judge*.

DECISION ON APPEAL

STATEMENT OF THE CASE

The Patent Examiner rejected claims 1, 2, 5, 6, 9, 11, 15-17, 19-21, and 23-34. The Appellants appeal therefrom under 35 U.S.C. § 134(a). We have jurisdiction under 35 U.S.C. § 6(b).

INVENTION

The Appellants describe the invention at issue on appeal as follows.

> [A] performance bottleneck for [Java Server Pages] JSP-generated Java Servlets is addressed by the present invention, in which the literal strings for the markup text is stored, not in Java literal strings, but in a resource file generated for each Java Servlet. In one implementation, the JSP-generated Java Servlet is configured to contain a [sic] inner class that includes initialization code that reads the markup text from the resource file and copies the markup text into static Java char arrays. The inner class is then "hotloaded," in which the class instructions and pre-initialized static variables are loaded into a shared, read-only memory that is globally available to different user sessions on the web server without synchronization. After a class is hotloaded, the class's instructions and pre-initialized static variables are available, without synchronization, to Java applications. By avoiding the use of Java literal strings, which require interning and synchronization, access to the static markup text of JSP-generated Java Servlets is much faster, which improves the performance of Java Servlets generated from Java Server Pages.

(Spec. 4.)

ILLUSTRATIVE CLAIM

1. A computer-implemented method of dynamically generating web pages, said method comprising:

analyzing a page that includes markup text and a set of code instructions executable on a server;

extracting the markup text from the page;

generating a servlet class for the page based on the set of code instructions, wherein the servlet class does not include the markup text;

loading a copy of the markup text into shared memory;

in response to each request of a plurality of requests for the page from a plurality of clients, performing the steps of

instantiating a distinct instance of the servlet class on the server, wherein instantiating each instance of the servlet class does not create another copy of the markup text;

executing said distinct instance of the servlet class, wherein execution of each instance of the server class generates a compiled page based on the copy of the markup text that resides in shared memory, and the set of code instructions; and

sending the compiled page to a client that requested the page.

PRIOR ART

Agrawal	US 2002/0004813 A1	Jan. 10, 2002
Claussen	US 6,675,354 B1	Jan. 6, 2004

REJECTION

Claims 1, 2, 5, 6, 9, 11, 15-17, 19-21, and 23-34 stand rejected under 35 U.S.C. § 103(a) as being unpatentable over Agrawal and Claussen.

CLAIM GROUPING

Based on the Appellants' arguments, we will decide the appeal of claims 1, 2, 5, 6, 9, 11, 15-17, 19-21 based on claim 1 alone, and the appeal of claims 23-34 based on claim 23 alone. *See* 37 C.F.R. § 41.37(c)(1)(vii).

CLAIMS 1, 2, 5, 6, 9, 11, 15-17, AND 19-21

The Examiner concludes that "[i]t would have been obvious to one of ordinary skill in the pertinent art at the time the invention was made to incorporate the teaching of Claussen into that of Agrawal for the inclusion of a servlet class." (Ans. 5.) The Appellants argue that "[t]here is no motivation to combine the two references." (Appeal Br. 12.) They also argue that "[a] typical servlet class and a typical instance of the servlet classes, both described in *Claussen*, are different from Applicants' servlet class and Applicants' instance because a typical class and instance are not created to handle a 'markup text' without making 'another copy of the markup text.' (Claim 1)[.]" (Reply Br. 5.)

ISSUE

Therefore, the issue before us is whether the Appellants have shown error in the Examiner's combination of teachings from Agrawal and Claussen and her finding that the combined teachings of these references would have suggested instantiating an instance of a servlet class without creating another copy of associated markup text.

LAW

The presence or absence of a reason "to combine references in an obviousness determination is a pure question of fact." *In re Gartside*, 203 F.3d 1305, 1316 (Fed. Cir. 2000). A reason to combine teachings from the prior art "may be found in explicit or implicit teachings within the references themselves, from the ordinary knowledge of those skilled in the art, or from the nature of the problem to be solved." *WMS Gaming Inc. v. Int'l Game*

Tech., 184 F.3d 1339, 1355 (Fed. Cir. 1999) (citing *In re Rouffet*, 149 F.3d 1350, 1355 (Fed. Cir. 1998)).

"The test for obviousness is what the combined teachings of the references would have suggested to one of ordinary skill in the art." *In re Young*, 927 F.2d 588, 591 (Fed. Cir. 1991) (citing *In re Keller*, 642 F.2d 413, 425 (CCPA 1981)). "Non-obviousness cannot be established by attacking references individually where the rejection is based upon the teachings of a combination of references." *In re Merck & Co.*, 800 F.2d 1091, 1097 (Fed. Cir. 1986) (citing *Keller*, 642 F.2d at 425). In determining obviousness, furthermore, a reference "must be read, not in isolation, but for what it fairly teaches in combination with the prior art as a whole." *Id.*

FINDINGS OF FACT ("FFS")

1. Agrawal describes its invention as follows.

> A method of servicing a request for a document over a computer network includes independently caching portions of pages called blocks. Each block includes a reference to a data source and code that is adapted to access the data source and to format the data accessed from the data source. When a request for a page is received over a computer network, one or more of the plurality of blocks defined in the script of the requested document may be retrieved from a cache memory. Any block that is not found in the cache memory is dynamically generated and a copy thereof is stored in the cache memory. The requested page may then be assembled from the page blocks retrieved from the cache memory and/or the dynamically generated page blocks.

(Abstract.)

2. Claussen explains that "major technologies have attempted to supplement, if not replace, dynamically generated HTML, i.e. database or CGI scripts used to generate a web page on the fly. These technologies [include] Microsoft's active server page (ASP)[and] Sun Microsystems's Java server page (JSP) . . . ." (Col. 1, l. 66 – col. 2, l. 3.) The latter reference includes the more specific description of these technologies.

> They provide for the generation and serving of dynamic web page content by enabling a page creator to write HTML and then to embed pure programming logic inside the page markup. Microsoft's ASP and Sun's JSP are very similar in that they both are essentially web templates that enable given code (e.g., code written in Java) to be embedded in static HTML to be served in response to a client browser request. In an illustrative example, a server (and, in particular, a Java runtime servlet) responds to a client jsp request as follows: the servlet retrieves a flat file corresponding to the requested page, translates that file into a Java servlet, compiles the servlet, class loads the servlet, and then invokes the servlet to cause given (e.g., customized) web content to be returned to the requesting browser.

(Col. 2, ll. 5-18.)

3. Claussen's "FIG. 2 is a high level flowchart illustrating a servlet generation routine . . . ." (Col. 3, ll. 63-64.)

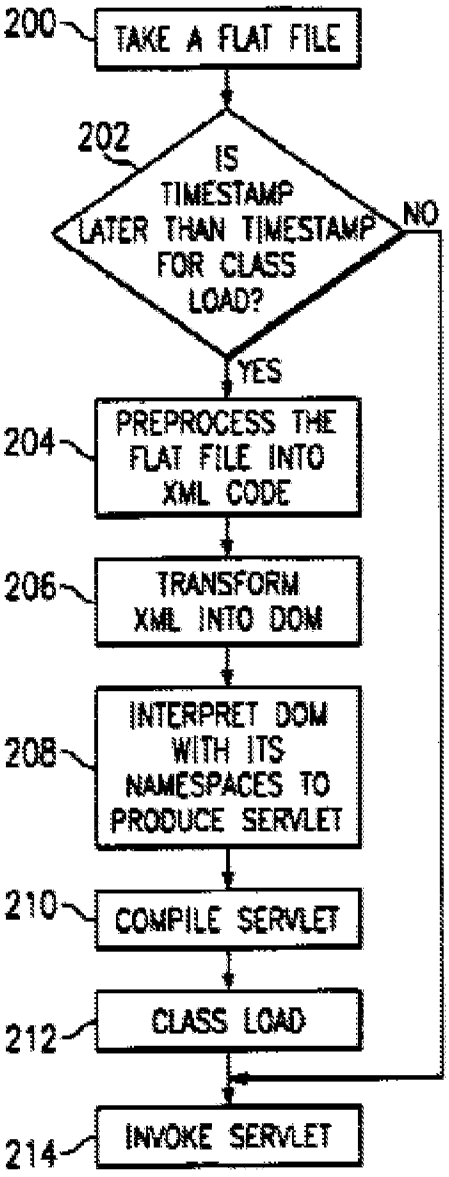

FIG. 2

More specifically, "FIG. 2 illustrates how a flat web page file is processed according . . . to generate a servlet." (Col. 5, ll. 3-4.)

ANALYSIS

Agrawal services a request for a document over a computer network. (FF 1.) In doing so, the reference dynamically generates blocks of the document that are unavailable in a cache. (*Id.*) For its part, Claussen describes Microsoft's ASP and Sun's JSP and explains that these major technologies are designed to supplement dynamically-generated web pages. (FF 2.) Furthermore, the Appellants describe their own method of generating a servlet. (FF 3.) We find that the fact that ASP and JSP, and presumably the Appellants' method, were designed to supplement dynamically generated web pages would have provided a reason to combine teachings of Agrawal and Claussen.

Accordingly, rejection at issue is based on what the combined teachings of Agrawal and Claussen would have suggested to one of ordinary skill in the art. The Appellants admit that "the *Claussen* reference describes a typical mechanism used to generate typical servlet classes . . . ." (Reply Br. 5.) Furthermore, the latter reference teaches that ASP and JSP class load a servlet (FF 2), as does Claussen's own method. (FF 3, Step 212.)

As mentioned *supra*, however, the Appellants argue that the latter reference does not mention generating servlet classes without making another copy of markup text. This individual attack on Claussen cannot establish non-obviousness.

In servicing a request for a document over a computer network, Agrawal retrieves blocks defined in the script of the requested document

from a cache memory (FF 1), without creating another copy thereof. When this teaching was combined with those of Claussen regarding class loading a servlet, we agree with the Examiner's finding that the combined teachings of Agrawal and Claussen would have suggested instantiating an instance of a servlet class without creating another copy of associated markup text.

CONCLUSION

Based on the aforementioned facts and analysis, we conclude that the Appellants have shown no error in the Examiner's finding that the combined teachings of Agrawal and Claussen would have suggested instantiating an instance of a servlet class without creating another copy of associated markup text.

CLAIMS 23-34

The Examiner finds that "Claussen further teaches wherein the servlet class includes an inner class (see at least co1.12:55 - co1.13:10)." (Ans. 7.) The Appellants argue "that the cited portions of Claussen refer to techniques for using multiple scripting languages to generate pages and it does not describe generating an inner class for a servlet class as recited in Claim 23." (App. Br. 13.)

ISSUE

Therefore, the issue before us is whether the Appellants have shown error in the Examiner's finding that a servlet class instantiated by Claussen includes an inner class.

LAW

"[T]he examiner bears the initial burden, on review of the prior art or on any other ground, of presenting a *prima facie* case of unpatentability." *In re Oetiker*, 977 F.2d 1443, 1445 (Fed. Cir. 1992). "'On appeal to the Board, an applicant can overcome a rejection by showing insufficient evidence of *prima facie* obviousness or by rebutting the *prima facie* case with evidence of secondary indicia of nonobviousness.'" *In re Kahn*, 441 F.3d 977, 985-86 (Fed. Cir. 2006) (quoting *In re Rouffet*, 149 F.3d 1350, 1355 (Fed. Cir. 1998)).

FINDING OF FACT

4. Claussen discloses a "routine, called jsp:block, [which] enables page developers to use multiple scripting languages in the same page. As will be seen, this enables people with different skill sets to add value to the same page." (Col. 12, ll. 55-59.)

5. The Examiner makes the following findings about the description of the routine.

> [The passage] explicitly discloses inserting methodDefinition element as a child (inner element) of the root element (outer element) of the HTML document. The same passage also discloses embedding one scripting language (i.e., inner element) within another scripting language (i.e., outer element). Needless to say, two elements from two different scripting languages must be defined differently (having different classes). Thus, a class element that is embedded in another class element (e.g., servlet class) clearly teaches the "inner class".

(Answer 12.)

ANALYSIS

Claussen discloses a routine that enables page developers to use multiple scripting languages in the same page; this enables people with different skill sets to add value to the same page. (FF 4.) The Examiner finds that this routine involves inserting an element as a child (inner element) of a root element (outer element) of an HTML document and embedding one scripting language (i.e., inner element) within another scripting language (i.e., outer element). (FF 5.) These findings are uncontested.

The Examiner also finds that two elements from two different scripting languages must feature different classes. (*Id.*) This finding is also uncontested. For example, an element from a JSP scripting language would have been generated from a different class than an element from an ASP scripting language. We agree with the Examiner that a class element that is inserted or embedded in another class element teaches an inner class.

CONCLUSION

Based on the aforementioned facts and analysis, we conclude that the Appellants have shown no error in the Examiner's finding that a servlet class instantiated by Claussen includes an inner class.

DECISION

We affirm the rejection of claims 1-2, 5-6, 9, 11, 15-17, 19-21, and 23-34.

No time for taking any action connected with this appeal may be extended under 37 C.F.R. § 1.136(a)(1). *See* 37 C.F.R. § 1.136(a)(1)(iv).

AFFIRMED

rwk

HICKMAN PALERMO TRUONG & BECKER/ORACLE
2055 GATEWAY PLACE
SUITE 550
SAN JOSE, CA 95110-1083